PH DE 030147

Europäisches Patentamt | European Patent Office | Office européen des brevets

PRIORITY DOCUMENT

SUBMITTED OR TRANSMITTED IN COMPLIANCE WITH RULE 17.1(a) OR (b)

Bescheinigung Certificate Attestation

Die angehefteten Unterlagen stimmen mit der ursprünglich eingereichten Fassung der auf dem nächsten Blatt bezeichneten europäischen Patentanmeldung überein.

The attached documents are exact copies of the European patent application described on the following page, as originally filed.

Les documents fixés à cette attestation sont conformes à la version initialement déposée de la demande de brevet européen spécifiée à la page suivante.

IB/04/52672

Patentanmeldung Nr. Patent application No. Demande de brevet n°

03104686.5

Der Präsident des Europäischen Patentamts;
Im Auftrag

For the President of the European Patent Office

Le Président de l'Office européen des brevets
p.o.

R C van Dijk

Anmeldung Nr:
Application no.: 03104686.5
Demande no:

Anmeldetag:
Date of filing: 15.12.03
Date de dépôt:

Anmelder/Applicant(s)/Demandeur(s):

Philips Intellectual Property & Standards
GmbH
Steindamm 94
20099 Hamburg
ALLEMAGNE
Koninklijke Philips Electronics N.V.
Groenewoudseweg 1
5621 BA Eindhoven
PAYS-BAS

Bezeichnung der Erfindung/Title of the invention/Titre de l'invention:
(Falls die Bezeichnung der Erfindung nicht angegeben ist, siehe Beschreibung.
If no title is shown please refer to the description.
Si aucun titre n'est indiqué se referer à la description.)

Nutzungsberechtigungseinrichtung für sicherheitsrelevante Anwendungen

In Anspruch genommene Priorität(en) / Priority(ies) claimed /Priorité(s) revendiquée(s)
Staat/Tag/Aktenzeichen/State/Date/File no./Pays/Date/Numéro de dépôt:

Internationale Patentklassifikation/International Patent Classification/
Classification internationale des brevets:

G07C9/00

Am Anmeldetag benannte Vertragstaaten/Contracting states designated at date of filing/Etats contractants désignées lors du dépôt:

AT BE BG CH CY CZ DE DK EE ES FI FR GB GR HU IE IT LU MC NL
PT RO SE SI SK TR LI

BESCHREIBUNG

Nutzungsberechtigungseinrichtung für sicherheitsrelevante Anwendungen

Die Erfindung betrifft eine Nutzungsberechtigungseinrichtung für sicherheitsrelevante Anwendungen, insbesondere die Zugangskontrolle zu Sicherheitsbereichen oder die Sicherung von Fahrzeugen, mit einer benutzerseitigen Schlüsseleinheit zur Erzeugung von aufeinanderfolgenden und voneinander unterschiedlichen Benutzercodeinformationen, die eine durch wiederholte Anwendung einer Einwegfunktion $F(v_i, const)$ erzeugte Sequenz von aufeinanderfolgenden Funktionswerten $v_{i+1} = F(v_i, const)$ für $i = 0,...,N$ aufweisen, welche in gegenüber der Sequenzbildung umgekehrter Reihenfolge zur Bildung der aufeinanderfolgenden Benutzercodeinformationen verwendet werden, und einer anwendungsseitigen Verarbeitungseinheit zum Bestimmen einer von der von der Schlüsseleinheit erhaltenen Benutzercodeinformation abhängigen Ist-Berechtigungsinformation und zum Ausführen eines Nutzungsberechtigungsprüfvorganges durch Vergleichen dieser Ist-Berechtigungsinformation mit einer anwendungsseitig vorliegenden Soll-Berechtigungsinformation sowie zur Erzeugung einer Nutzungsfreigabeinformation in Abhängigkeit vom Ergebnis des Vergleiches, wobei die Soll-Berechtigungsinformation einen Funktionswert v_i aufweist, der von derjenigen Benutzercodeinformation übernommen wurde, die im zuletzt positiv verlaufenden Nutzungsberechtigungsvorgang verarbeitet wurde.

Eine solche Nutzungsberechtigungseinrichtung ist beispielsweise aus der DE 44 11 449 C1 bekannt, wobei die dort beschriebene Nutzungsberechtigungseinrichtung zur Fahrzeugsicherung vorgesehen und Teil einer Wegfahrsperre ist.

Eine derartige Nutzungsberechtigungseinrichtung arbeitet nach einem sogenannten Wechselcodeverfahren, bei welchem die Sicherheit gegenüber einer unberechtigten Benutzung der sicherheitsrelevanten Anwendung nach Abhören der gesendeten Codeinformationen dadurch erhöht wird, dass die Codeinformation bei jedem

Prüfvorgang der Nutzungsberechtigung, auch Authentifizierungsvorgang genannt, wechselt. Dieser Codewechsel lässt sich mit Hilfe einer unidirektionalen Codeinformationsübertragung nur von der Schüssel- zur Anwendungsseite dadurch realisieren, dass sowohl auf der Schlüssel- als auch auf der Anwendungsseite eine geheime Information eine über Basiszahl bzw. einen Startwert und ein Algorithmus abgelegt sind, nach welchem aufeinander folgende Codeinformationen aus dem Startwert ableitbar sind. Auf diese Weise kann durch jeweiligen Vergleich der anwendungsseitig erzeugten Codeinformation mit der schlüsselseitig gesendeten Codeinformation auf der Seite der Anwendung die Benutzungsberechtigung überprüft werden.

Für die unidirektionale Codeinformationsübertragung ist eine Einwegfunktion $F(v_i, const)$ vorgegeben, die nicht oder nur mit erheblichem zeitlichen und/oder finanziellen Aufwand zu invertieren ist. Hierzu kommen insbesondere eine hochgradig nichtlineare Boolesche Funktion oder eine Hash-Funktion in Frage. Ausgehend von einem Startwert v_0 lassen sich mit Hilfe

$$v_{i+1} = F(v_i, const),\ i = (0,...,N) \qquad (1)$$

iterativ neue Funktionswerte berechnen, von denen nur der jeweils letzte Funktionswert ein variabler Parameter ist, wobei der Parameter const eine Konstante und/oder der Funktionswert für einen bestimmten Index i und/oder ein nur der Schlüsseleinheit und der Zugangskontrolleinheit sein kann. Für die Nutzungsberechtigungseinheit werden die Funktionswerte in absteigender Reihenfolge von v_{max} bis v_0 benutzt. Eine geeignete Implementierung benötigt die Werte v_0 und *const*, um daraus alle Funktionswerte zwischen v_0 und v_{max} in aufsteigender Reihenfolge berechnen zu können. Dabei müssen die Startwerte v_0 und *const* gemeinsam sowohl auf der Seite der Schlüsseleinheit als auch auf der Seite der sicherheitsrelevanten Anwendung bekannt sein.

An dieser Stelle sei der Vollständigkeit halber noch darauf hingewiesen, dass die Nutzungsberechtigungseinrichtung der zuvor beschriebenen Art nicht nur auf eine Fahrzeugsicherung beschränkt ist; vielmehr bilden ein wichtiges Einsatzgebiet beispielsweise Zugangskontrollsysteme zu allen Arten von Sicherheitsbereichen, bei denen sich der Schlüssel gegenüber dem Schloss autorisieren muss.

Es ist im Allgemeinen nicht zweckmäßig für eine große Zahl von vorgesehenen Autorisierungen (z.B. 100000) alle dazu benötigten Werte innerhalb der Schlüsseleinheit zu speichern. Somit muss die Schlüsseleinheit über ein Verfahren verfügen, welches ihr ermöglicht, mit Kenntnis der Funktion F aus Gleichung (1) und eines Startwertes v_0 oder mehrerer Werte v_i den aktuell benötigten Ausgabewert zu berechnen. Einfache Ansätze einer Implementierung der zuvor beschriebenen Nutzungsberechtigungseinheit leiden im Allgemeinen unter dem Nachteil, dass der Aufwand an Rechenzeit und/oder Speicherplatz für die Berechnung der einzelnen aufeinanderfolgenden Ausgabewerte während der Laufzeit erheblich schwankt, was den Einsatz in Systemen mit beschränkten Ressourcen erschwert oder sogar verhindert. Für derartige Systeme ist eine Technik mit skalierbarem Aufwand hinsichtlich Speicherplatz und Rechenzeit unabdingbar. Zusätzlich sollte das System im Betrieb eine vorher genau zu bestimmende Zeitspanne zwischen der Übermittlung zweier aufeinander folgender Ausgabewerte haben, um diverse Systemparameter darauf abstimmen zu können.

Deshalb schlägt die Erfindung vor, bei einer Nutzungsberechtigungseinrichtung der eingangs genannten Art eine bestimmte Anzahl von Stufen G vorzusehen, wobei in jeder Stufe eine bestimmte Anzahl von iterativen Funktionswertberechnungen mittels der Einwegfunktion $F(v_i, const)$ durchführbar ist, wobei die Anzahl der Stufen $G = \lceil L(N)/b \rceil$ ist, wobei N der Startwert, $L(N)$ die Anzahl der notwendigen Bits zur Darstellung von N im Dualsystem und b die Basis ist, welche zur Festlegung der Anzahl der Stufen und der Anzahl der in jedem Iterationsschritt notwendigen

Funktionswertberechnungen benutzt wird ($\lceil x \rceil$ ist die kleinste ganze Zahl größer oder gleich x).

Die Erfindung eignet sich für die Realisierung des zuvor beschriebenen unirektionalen Verschlüsselungsverfahrens besonders vorteilhaft auf Geräten mit beschränkten Ressourcen an flüchtigem und/oder nicht-flüchtigem Speicherplatz z.B. im RAM und EEPROM und/oder Rechenleistung wie z.B. einen programmierbaren oder nicht-programmierbaren sogenannten embedded Controller. Insbesondere bei Verwendung der Erfindung auf sogenannten embedded Low-Cost-Controllern (z.B. für "remote keyless entry" in der Automobilbranche) machen sich die geringe Entwicklungszeit der Implementierung, die Konstanz des Verfahrens hinsichtlich des Bedarfes an Speicherplatz während der Laufzeit, die Skalierbarkeit hinsichtlich Speicherbedarf und Ausführungszeit und das robuste Verhalten gegenüber Unterbrechungen während der Laufzeit (z.B. Unterbrechung der Stromversorgung bei induktiv betriebenen Schlüsseleinheiten) positiv bemerkbar. Das Gleiche gilt auch für eine Implementierung der Erfindung als Hard-Wired-Logic-Device.

Vorzugsweise ist für jede Stufe eine Stützstelle $s(i)$ mit $i=(1,...,G)$ vorgesehen. Hierbei werden einige Funktionswerte als sogenannte Stützstellen eingerichtet, welche entweder systembedingt schon vorgegeben sind oder von der Schlüsseleinheit selbst mittels Iteration berechnet werden, bevor der Algorithmus startet. Mittels weiterer Stützstellen pro Stufe oder nur für bestimmte Stufen kann der Rechenaufwand verringert werden.

Zweckmäßigerweise ergeben sich die Werte der Stützstellen $s(i)$ aus der Gleichung

$$s(i)=N-\sum_{j=1}^{i}\left(2^{b}\right)^{j} \qquad (2)$$

wobei N der in Gleichung (1) definierte Startwert der Ausgabewerte ist. Gewöhnlich können für negative Indizes keine Funktionswerte berechnet werden, was die

Anwendung der Gleichung (2) auf Stützstellen mit positivem Index beschränkt.

Vorzugsweise sollte zu einer vorgegebenen Anzahl von Stützstellen der Parameter b so angepasst werden, dass sich ein Minimum an Funktionswertberechnungen pro Nutzungsberechtigung ergibt.

Gewöhnlich sollte in jeder Stufe, ausgehend von der jeweiligen Stützstelle $s(i)$, eine bestimmte Anzahl von Funktionswerten in absteigender Reihenfolge berechnet und als Zwischenwerte abgespeichert werden. Dabei sollte sukzessive in einer Stufe ein Zwischenwert zur Stützstelle dieser Stufe zurückgesetzt werden, nachdem dieser Zwischenwert als neue Stützstelle in die nächstkleinere Stufe übertragen worden ist.

Um beliebige Startwerte N zuzulassen, können entweder die entsprechend benötigten Zwischenwerte mitgeliefert oder als Startwert $N=\left(2^b\right)^G$, indem das Verfahren mit $\left(2^b\right)^G$ gestartet und bis zum Index $i=N$ durchgeführt werden kann, vorgegeben werden.

Alternativ ist es aber auch denkbar, dass für den Startwert

$$N \in \left\{\left(2^b\right)^{G-1},\ldots,\left(2^b\right)^G-1\right\} \qquad (3)$$

gilt, was die uneingeschränkte Anwendung von Gleichung (2) für die Berechnung der Stützstellen erlaubt. Dies lässt sich durch geeignete Vorgaben des Systems erreichen.

Zweckmäßigerweise sollten pro Stufe mehrere Zwischenspeicherungen zur Speicherung von aus den Funktionswertberechnungen berechneten Zwischenwerten vorgesehen sein. Um die Schreiboperationen beim Speichern der Zwischenwerte zu verringern, können pro Stufe mehrere Zwischenspeicher vorgesehen sein, die rotierend beschrieben und gelesen werden und somit als FIFO-Speicher ausgebildet sind.

Nachfolgend wird ein bevorzugtes Ausführungsbeispiel der Erfindung anhand der beiliegenden Figuren näher erläutert. Es zeigen:

Fig. 1 ein schematisches Blockschaltbild einer bevorzugten Ausführung des erfindungsgemäßen Nutzungsberechtigungssystems;

Fig. 2 eine schematische Darstellung der Anordnung von Stützstellen und Zwischenwerten und der Abstände zwischen den einzelnen Stützstellen nach der Initialisierung des Nutzungsberechtigungssystems von Figur 1;

Fig. 3 eine schematische Darstellung der Anordnung von Stützstellen und Zwischenwerten und der Abstände zwischen den einzelnen Stützstellen bei Ausgabe des achten Wertes; und

Fig. 4 eine schematische Darstellung der Anordnung von Stützstellen und Zwischenwerten und der Abstände zwischen den einzelnen Stützstellen bei Ausgabe des neunten Wertes.

Nachfolgend wird eine bevorzugte Ausführung eines Nutzungsberechtigungssystems, das in Figur 1 schematisch im Blockschaltbild dargestellt ist, anhand eines darin implementierten Verfahrens beschrieben, welches zwischen der Ausgabe zweier aufeinanderfolgender Werte einen konstanten Bedarf an Arbeitsspeicher (z.B. RAM und/oder EEPROM) zur Folge hat. Dabei ist mittels der Anzahl von benutzten Stützstellen skalierbar, ob bei erhöhtem Arbeitsspeicherbedarf eine reduzierte Ausführungszeit vonnöten ist oder umgekehrt, während der Bedarf an Programmspeicher (ROM) hiervon nahezu unberührt bleibt.

Wie bereits zuvor beschrieben, ist eine Einwegfunktion $F(v_i, const)$ vorgegeben, die nicht bzw. nur mit erheblichem Aufwand zu invertieren ist (z.B. eine hochgradig nichtlineare Boolesche Funktion oder eine Hash-Funktion). Ausgehend von einem Startwert v_0 lassen sich iterativ neue Funktionswerte v_i berechnen mittels

$$v_{i+1} = F(v_i, const),\ i = (0,...,N)$$

(1)

wobei nur der jeweils letzte Funktionswert ein variabler Parameter ist. Der Parameter *const* kann eine Konstante und/oder der Funktionswert für einen bestimmten Index i oder ein nur dieser Nutzungsberechtigungseinrichtung bekannter Wert sein.

Die Aufgabe besteht darin, die Funktionswerte in absteigender Reihenfolge von v_{max} bis zu v_0 auszugeben. Da die inverse Funktion von F nicht bekannt ist, benötigt eine Implementierung v_0 und *const* um daraus alle Werte zwischen v_0 und v_{max} berechnen zu können. Die hier beschriebene Methode benötigt einige Funktionswerte, als sog. Stützstellen, welche entweder schon gegeben sind oder mittels Iteration aus den gegebenen Startwerten berechnet werden, bevor der hier beschriebene Algorithmus startet.

Die Funktionswerte v_i sind für Indizes i von N bis 0 auszugeben, wobei, wie oben beschrieben, nur jeweils $v_{i+1} = F(v_i, const)$ berechnet werden kann. $L(N)$ ist die Anzahl der notwendigen Bits zur Darstellung von N im Dualsystem und b die Basis, wobei die Basis bestimmt, wie viele Stützstellen unbedingt notwendig sind und wie viele iterative Funktionsberechnungen pro Stufe maximal notwendig sind. Da die gesamte Implementierung auf dem Index i aufbaut, lässt sich der Wert der Basis b auch als Zahl der konsekutiven Bits in i interpretieren: Die Bits 0 bis b bilden den Zähler für die Stufe 1, die Bits $b+1$ bis $2b$ den Zähler für Stufe 2 usw. Die Anzahl der Stufen selbst berechnet sich aus

$$G = \left\lceil \frac{L(N)}{b} \right\rceil \qquad (3)$$

was gleichzeitig auch der Zahl der unbedingt notwendigen Stützstellen entspricht, da jede Stufe mindestens eine Stützstelle benötigt. Diese Stützstellen werden nachfolgend als $s(i)$ mit $i = (1,...,G)$ für die jeweiligen Stufen i bezeichnet.

Für die Werte der Stützstellen selbst lassen sich zwei Fälle unterscheiden:

Fall 1: Für $N=\left(2^b\right)^G$ gilt $s(g)=N-\sum_{j=1}^{g}\left(2^b\right)^j$ mit $g=(1,...,G)$. Damit ergibt sich, dass $s(G)$ negativ ist. Falls dieser Wert nicht berechnet werden kann, ergibt sich eine kleine Abwandlung für den nachfolgend beschriebenen Algorithmus, welche aber nicht den grundsätzlichen Ablauf berührt.

Fall 2: Für $N\in\left\{\left(2^b\right)^{G-1},...,\left(2^b\right)^G-1\right\}$ ist es hinreichend, den nachfolgend beschriebenen Algorithmus bis zum Erreichen des Index N zu simulieren und sich die dann abgespeicherten Zwischenwerte und Stützstellen ausgeben zu lassen und als Initialisierungswerte zu benutzen.

Sei obiger Fall 1 gegeben mit $b=3$ und $N=\left(2^3\right)^4=4096$, dann ist $G=4$, und die Stützstellen sind unter Benutzung von Gleichung (3) gegeben durch:

$$s(1)=v_{N-8}=v_{4088}$$
$$s(2)=v_{N-72}=v_{4024}$$
$$s(3)=v_{N-584}=v_{3512}$$
$$s(4)=v_{N-4680}=v_{-584}$$

Dabei sei $s(4)$ als gegeben vorausgesetzt, da dieser aus v_0 nicht berechnet werden kann. Nach Ausgabe des ersten Wertes $v_{N=4096}$, hat der Zähler C den Wert $C=N-1=4095=\left(\underbrace{111}_{=c(4)}.\underbrace{111}_{=c(3)}.\underbrace{111}_{=c(2)}.\underbrace{111}_{=c(1)}\right)_b$, d.h. die Zähler $c(i)$ der einzelnen Stufen zur Basis $b=3$ sind jeweils $c(i)=(111)_b=7$.

Die zuvor beschriebene Anordnung der Stützstellen und der Abstände zwischen diesen einzelnen Stützstellen ist schematisch in Fig. 2 dargestellt und bildet gleichzeitig die Ausgangssituation für den nachfolgend beschriebenen beispielhaften Algorithmus.

Nun soll der zweite Ausgabewert ausgehend von der in Fig. 1 dargestellten Konfiguration, berechnet werden. Entsprechend der Zählerwerte $c(i)$ werden von den Stützstellen $s(i)$ durch iteratives Anwenden von $F(v_i)$ die Zwischenwerte $z(i)$ berechnet. Diese sind gegeben durch:

$$\begin{aligned} s(4) &= v_{-584} & z(4) &= v_{-577} & c(4) &= 7 \\ s(3) &= v_{3512} & z(3) &= v_{3519} & c(3) &= 7 \\ s(2) &= v_{4024} & z(2) &= v_{4031} & c(2) &= 7 \\ s(1) &= v_{4088} & z(1) &= v_{4095} & c(1) &= 7 \end{aligned}$$

$$C = 4095 = (111.111.111.111)_b$$

Dabei ist $z(1)$ der gesuchte zweite Ausgabewert.

Um den dritten Ausgabewert zu berechnen, wiederholt sich das Ganze mit den Stufenzählern $c(4)=7$, $c(3)=7$, $c(2)=7$ und $c(1)=6$, die sich aus dem Zähler $C = 4094 = \left(\underbrace{111}_{=c(4)}.\underbrace{111}_{=c(3)}.\underbrace{111}_{=c(2)}.\underbrace{110}_{=c(1)} \right)_b$ ergeben. In allen Stufen außer Stufe 1 werden statt $s(i)$ als Startwerte die sich entsprechend geänderten Zwischenwerte $z(i)$ verwendet. In Stufe 1 wird dagegen immer $s(1)$ verwendet. Damit ergibt sich für die Zwischenwerte:

$$\begin{aligned} s(4) &= v_{-584} & z(4) &= v_{-570} & c(4) &= 7 \\ s(3) &= v_{3512} & z(3) &= v_{3526} & c(3) &= 7 \\ s(2) &= v_{4024} & z(2) &= v_{4038} & c(2) &= 7 \\ s(1) &= v_{4088} & z(1) &= v_{4094} & c(1) &= 6 \end{aligned}$$

$$C = 4094 = (111.111.111.110)_b$$

und in $z(1)$ steht der gewünschte Ausgabewert. Nach weiteren sechs berechneten Ausgabewerten gilt:

$$s(4)=v_{-584} \qquad z(4)=v_{-528} \qquad c(4)=7$$
$$s(3)=v_{3512} \qquad z(3)=v_{3568} \qquad c(3)=7$$
$$s(2)=v_{4024} \qquad z(2)=v_{4080} \qquad c(2)=7$$
$$s(1)=v_{4088} \qquad z(1)=v_{4088} \qquad c(1)=0$$
$$C=4088=(111.111.111.000)_b$$

Somit sind die Stufen größer 1 für jeden Ausgabewert gemäß ihrem Stufenzähler $c(i)$ um jeweils sieben Iterationen der Funktion F vorangekommen. In Stufe 1 ist der jeweilige Ausgabewert berechnet worden, indem der aktuelle Zähler $c(1)$ als Anzahl der Iterationen von F ausgehend von $s(1)$ herangezogen wurde. Dieser Zustand ist in Fig. 3 illustriert.

Um nun den nächsten Ausgabewert für $C=4087=(111.111.110.111)_b$ berechnen zu können, muss $s(1)$ nach Ausgabe von v_{4088} ersetzt werden. Das Verfahren ist so konstruiert, dass der gesuchte Wert automatisch in $z(2)$ steht, sobald der Zähler $c(1)=0$ ist. Somit wird $s(1):=z(i)$ gesetzt und $z(2):=s(2)$ wieder auf den ursprünglichen Wert zurückgesetzt, so dass sich nach der Berechnung des nächsten Ausgabewertes folgendes ergibt:

$$s(4)=v_{-584} \qquad z(4)=v_{-521} \qquad c(4)=7$$
$$s(3)=v_{3512} \qquad z(3)=v_{3575} \qquad c(3)=7$$
$$s(2)=v_{4024} \qquad z(2)=v_{4030} \qquad c(2)=6$$
$$s(1)=v_{4080} \qquad z(1)=v_{4087} \qquad c(1)=7$$
$$C=4087=(111.111.110.111)_b$$

Dieser Zustand nach dem ersten Unterlauf ist in Fig. 4 illustriert.

Nun können die zuvor beschriebenen Schritte zur Ausgabe der Werte von $C=4087$ bis $C=4080$ sukzessive wiederholt werden, wobei $c(4)=7$, $c(3)=7$ und $c(2)=6$ gilt. Für $C=4080$ ergibt sich folgender Stand:

$$s(4)=v_{-584} \quad z(4)=v_{-472} \quad c(4)=7$$
$$s(3)=v_{3512} \quad z(3)=v_{3617} \quad c(3)=7$$
$$s(2)=v_{4024} \quad z(2)=v_{4066} \quad c(2)=6$$
$$s(1)=v_{4080} \quad z(1)=v_{4080} \quad c(1)=0$$
$$C=4080=(111.111.110.000)_b$$

Nach jedem Unterlauf von $c(1)$ wird die Ersetzung $s(1):=z(i)$ und $z(2):=s(2)$ durchgeführt. Beim Zählerstand $C=4032$ gilt:

$$s(4)=v_{-584} \quad z(4)=v_{-136} \quad c(4)=7$$
$$s(3)=v_{3512} \quad z(3)=v_{3960} \quad c(3)=7$$
$$s(2)=v_{4024} \quad z(2)=v_{4024} \quad c(2)=0$$
$$s(1)=v_{4032} \quad z(1)=v_{4032} \quad c(1)=0$$
$$C=4032=(111.111.000.000)_b$$

Damit ergibt sich im nächsten Schritt außer für den schon bekannten Unterlauf von $c(1)$ auch ein Unterlauf von $c(2)$. Konstruktionsbedingt findet sich der neue Wert für $s(2)$ im Zwischenwert $z(3)$ der nächsthöheren Stufe. Es wird dann entsprechend $s(2):=z(3)$ und $z(2):=s(2)$ sowie $s(1):=z(2)$ und $z(1):=s(1)$ gesetzt. Damit ergibt sich dann für $C=4031$:

$$s(4)=v_{-584} \quad z(4)=v_{-129} \quad c(4)=7$$
$$s(3)=v_{3512} \quad z(3)=v_{3518} \quad c(3)=6$$
$$s(2)=v_{3960} \quad z(2)=v_{3967} \quad c(2)=7$$
$$s(1)=v_{4024} \quad z(1)=v_{4031} \quad c(1)=7$$
$$C=4031=(111.110.111.111)_b$$

Der Übertrag eines Zwischenwertes als neue Stützstelle der nächstkleineren Stufe und das Rücksetzen des Zwischenwertes zur Stützstelle derselben Stufe setzt sich sukzessive fort. Der maximale Rechenaufwand ergibt sich im Wesentlichen aus der maximalen Anzahl von Funktionsberechnungen $v_{k+1}=F(v_k,const)$ zwischen zwei

aufeinanderfolgenden Ausgabewerten und somit aus der maximalen Anzahl der Werte in $c(i)$, also 28=4·7 in obigem Beispiel.

Damit beliebige Startwerte N verwendet werden können, müssen entweder die entsprechend benötigten Stützstellen $s(i)$ mitgeliefert oder kann das Verfahren mit $\left(2^b\right)^G$ gestartet und bis zum Index $i = N$ während einer Initialisierung des Systems durchgeführt werden, bevor es dann mit dem zuvor beschriebenen Verfahren weitergeht. Mittels weiterer Stützstellen pro Stufe oder nur für bestimmte Stufen kann außerdem der Rechenaufwand verringert werden. Zu einer gegebenen Anzahl möglicher Stützstellen kann ferner der Parameter b so angepaßt werden, dass nur ein Minimum an Funktionswertberechnungen pro Nutzungsberechtigung erforderlich wird, was auch die Gesamtzahl erforderlicher Funktionswertberechnungen des Systems während der gesamten Laufzeit minimiert. Schließlich sollte für Stützstellen mit negativem Index, zu denen keine Funktionswerte berechnet werden können, eine Abfrage auf $i = 0$ stattfinden.

Um die Schreiboperationen beim Speichern der Zwischenwerte zu verringern, können pro Stufe mehrere Zwischenspeicher vorgesehen sein, die rotierend beschrieben und gelesen werden und somit als FIFO-Speicher ausgebildet sind.

PATENTANSPRÜCHE

1. Nutzungsberechtigungseinrichtung für sicherheitsrelevante Anwendungen, insbesondere die Zugangskontrolle zu Sicherheitsbereichen oder die Sicherung von Fahrzeugen, mit

- einer benutzerseitigen Schlüsseleinheit zur Erzeugung von aufeinanderfolgenden und voneinander unterschiedlichen Benutzercodeinformationen, die eine durch wiederholte Anwendung einer Einwegfunktion $F(v_i, const)$ erzeugte Sequenz von aufeinanderfolgenden Funktionswerten $v_{i+1} = F(v_i, const)$ für $i = 0,...,N$ aufweisen, welche in gegenüber der Sequenzbildung umgekehrter Reihenfolge zur Bildung der aufeinanderfolgenden Benutzercodeinformationen verwendet werden; und
- einer anwendungsseitigen Verarbeitungseinheit zum Bestimmen einer von der von der Schlüsseleinheit erhaltenen Benutzercodeinformation abhängigen Ist-Berechtigungsinformation und zum Ausführen eines Nutzungsberechtigungsprüfvorganges durch Vergleichen dieser Ist-Berechtigungsinformation mit einer anwendungsseitig vorliegenden Soll-Berechtigungsinformation sowie zur Erzeugung einer Nutzungsfreigabeinformation in Abhängigkeit vom Ergebnis des Vergleiches, wobei die Soll-Berechtigungsinformation einen Funktionswert v_i aufweist, der von derjenigen Benutzercodeinformation übernommen wurde, die im zuletzt positiv verlaufenden Nutzungsberechtigungsvorgang verarbeitet wurde;

dadurch gekennzeichnet, dass

- eine bestimmte Anzahl von Stufen G vorgesehen ist, von denen in jeder Stufe eine bestimmte Anzahl von iterativen Funktionswertberechnungen mittels der Einwegfunktion $F(v_i, const)$ durchführbar ist, und
- die Anzahl der Stufen $G = \lceil L(N)/b \rceil$ ist, wobei N der Startwert, $L(N)$ die Anzahl der notwendigen Bits zur Darstellung von N im Dualsystem und b die Basis ist.

2. Einrichtung nach Anspruch 1,
dadurch gekennzeichnet,
dass für jede Stufe eine Stützstelle $s(i)$ mit $i = (1,...,G)$ vorgesehen ist.

3. Einrichtung nach Anspruch 2,
dadurch gekennzeichnet,
dass sich die Werte der Stützstellen $s(i)$ aus der Gleichung

$$s(i) = N - \sum_{j=1}^{i} (2^b)^j$$

ergeben.

4. Einrichtung nach Anspruch 2 oder 3,
dadurch gekennzeichnet,
dass für Stützstellen mit negativem Index keine Funktionswerte berechnet werden.

5. Einrichtung nach mindestens einem der Ansprüche 2 bis 4,
dadurch gekennzeichnet,
dass zu einer gegebenen Anzahl von Stützstellen der Parameter b so angepasst wird, dass sich ein Minimum an Funktionswertberechnungen ergibt.

6. Einrichtung nach mindestens einem der Ansprüche 2 bis 5,
dadurch gekennzeichnet,
dass in jeder Stufe, ausgehend von der jeweiligen Stützstelle $s(i)$, eine bestimmte Anzahl von Funktionswerten in absteigender Reihenfolge berechnet und als Zwischenwerte abgespeichert werden.

7. Einrichtung nach Anspruch 6,
dadurch gekennzeichnet,
dass sukzessive in einer Stufe ein Zwischenwert zur Stützstelle dieser Stufe zurückgesetzt wird, nachdem dieser Zwischenwert als neue Stützstelle in die nächstkleinere Stufe übertragen worden ist.

8. Einrichtung nach mindestens einem der vorangegangenen Ansprüche,
dadurch gekennzeichnet,
dass der Startwert $N = \left(2^b\right)^G$ ist.

9. Einrichtung nach mindestens einem der Ansprüche 1 bis 7,
dadurch gekennzeichnet,
dass für den Startwert $N \in \left\{\left(2^b\right)^{G-1}, \ldots, \left(2^b\right)^G - 1\right\}$ gilt.

10. Einrichtung nach mindestens einem der vorangegangenen Ansprüche, dadurch gekennzeichnet, dass pro Stufe mehrere Zwischenspeicherungen zur Speicherung von aus den Funktionswertberechnungen berechneten Zwischenwerten vorgesehen sind.

11. Einrichtung nach Anspruch 10, dadurch gekennzeichnet, dass die Zwischenspeicher FIFO-Speicher sind.

ZUSAMMENFASSUNG

Nutzungsberechtigungseinrichtung für sicherheitsrelevante Anwendungen

Beschrieben wird eine Nutzungsberechtigungseinrichtung für sicherheitsrelevante Anwendungen, insbesondere die Zugangskontrolle zu Sicherheitsbereichen oder die Sicherung von Fahrzeugen, mit einer benutzerseitigen Schlüsseleinheit zur Erzeugung von aufeinanderfolgenden und voneinander unterschiedlichen Benutzercodeinformationen, die eine durch wiederholte Anwendung einer Einwegfunktion $F(v_i, const)$ erzeugte Sequenz von aufeinanderfolgenden Funktionswerten $v_{i+1} = F(v_i, const)$ für $i = 0,...,N$ aufweisen, welche in gegenüber der Sequenzbildung umgekehrter Reihenfolge zur Bildung der aufeinanderfolgenden Benutzercodeinformationen verwendet werden, und einer anwendungsseitigen Verarbeitungseinheit zum Bestimmen einer von der von der Schlüsseleinheit erhaltenen Benutzercodeinformation abhängigen Ist-Berechtigungsinformation und zum Ausführen eines Nutzungsberechtigungsprüfvorganges durch Vergleichen dieser Ist-Berechtigungsinformation mit einer anwendungsseitig vorliegenden Soll-Berechtigungsinformation sowie zur Erzeugung einer Nutzungsfreigabeinformation in Abhängigkeit vom Ergebnis des Vergleiches, wobei die Soll-Berechtigungsinformation einen Funktionswert v_i aufweist, der von derjenigen Benutzercodeinformation übernommen wurde, die im zuletzt positiv verlaufenden Nutzungsberechtigungsvorgang verarbeitet wurde. Das Besondere der Erfindung besteht darin, dass eine bestimmte Anzahl von Stufen G mit mindestens einer Stützstelle und einem Zwischenwert vorgesehen ist, von denen in jeder Stufe eine bestimmte Anzahl von iterativen Funktionswertberechnungen mittels der Einwegfunktion $F(v_i, const)$ durchführbar ist wobei die Anzahl der Stufen mit $G = \lceil L(N)/b \rceil$ gegeben ist, mit N als Startwert, $L(N)$ die Anzahl der notwendigen Bits zur Darstellung von N im Dualsystem und b die Basis ist.

Fig.1

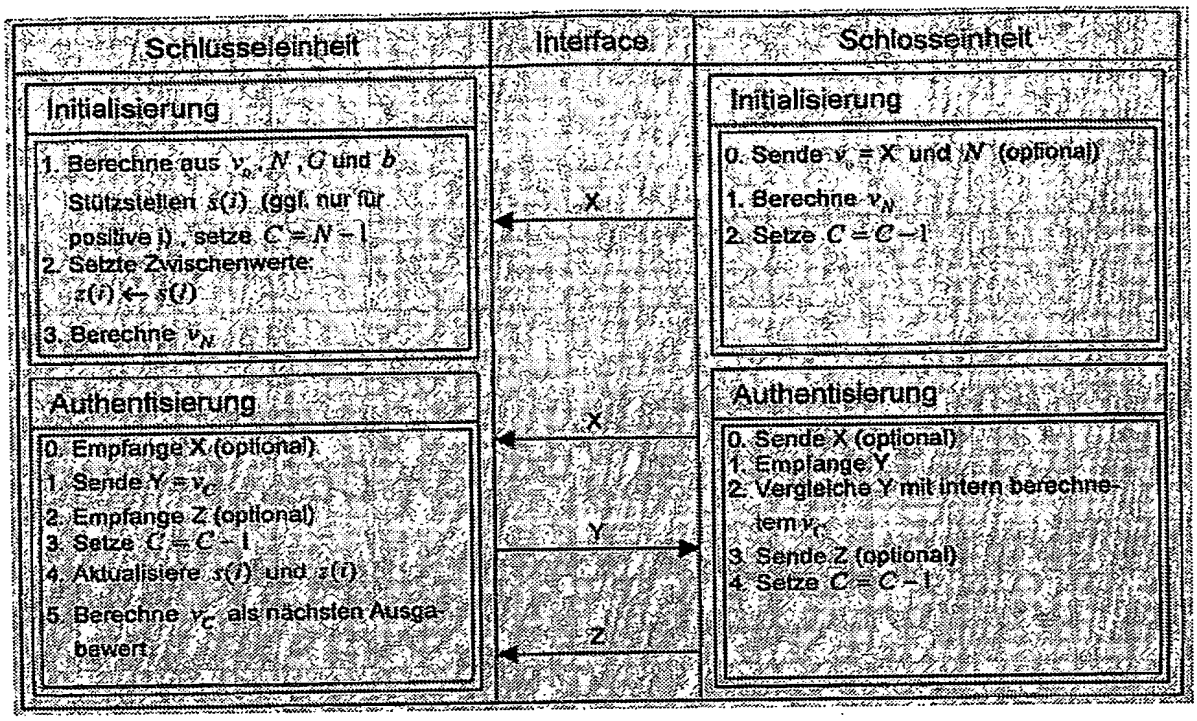
Schlusseleinheit
Interface
Schlosseinheit
Initialisierung
1. Berechne aus v_0, N, C und b Stützstellen $s(i)$ (ggf. nur für positive i), setze $C = N - 1$
2. Setzte Zwischenwerte: $z(i) \leftarrow s(i)$
3. Berechne v_N
X
Initialisierung
0. Sende $v_0 = X$ und N (optional)
1. Berechne v_N
2. Setze $C = C - 1$
Authentifisierung
0. Empfange X (optional)
1. Sende $Y = v_C$
2. Empfange Z (optional)
3. Setze $C = C - 1$
4. Aktualisiere $s(i)$ und $z(i)$
5. Berechne v_C als nächsten Ausgabewert
X
Y
Z
Authentisierung
0. Sende X (optional)
1. Empfange Y
2. Vergleiche Y mit intern berechnetem v_C
3. Sende Z (optional)
4. Setze $C = C - 1$

Fig. 1

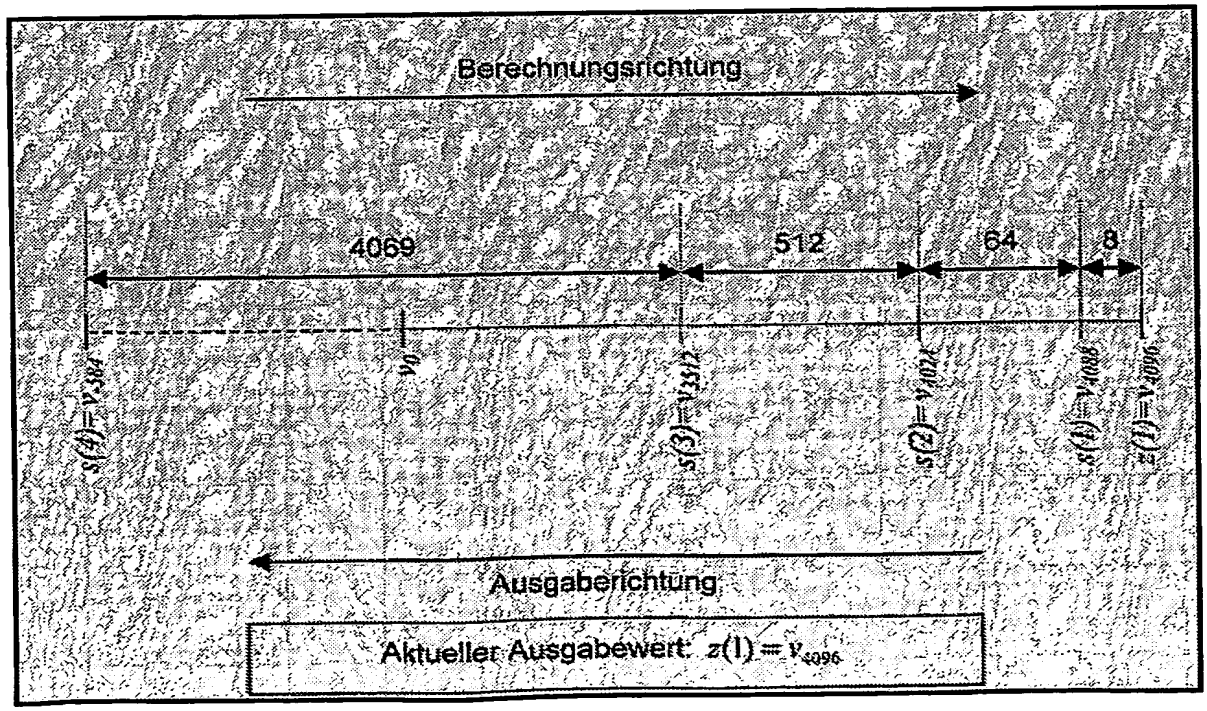
Berechnungsrichtung
4069
512
64
8
v_0
$s(4) = v_{3584}$
$s(3) = v_{3512}$
$s(2) = v_{4032}$
$s(1) = v_{4088}$
$z(1) = v_{4096}$
Ausgaberichtung
Aktueller Ausgabewert: $z(1) = v_{4096}$

Fig. 2

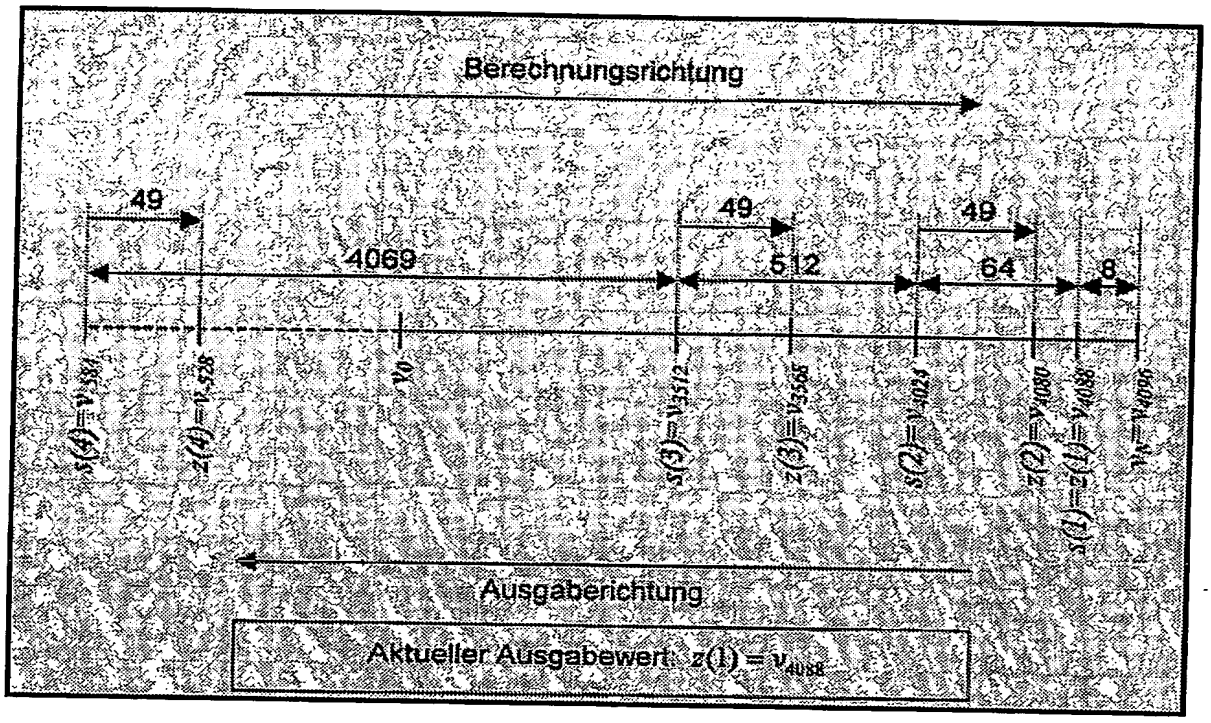
Berechnungsrichtung
49
4069
49
512
49
64
8
Ausgaberichtung
Aktueller Ausgabewert: $z(1) = v_{4088}$

Fig. 3

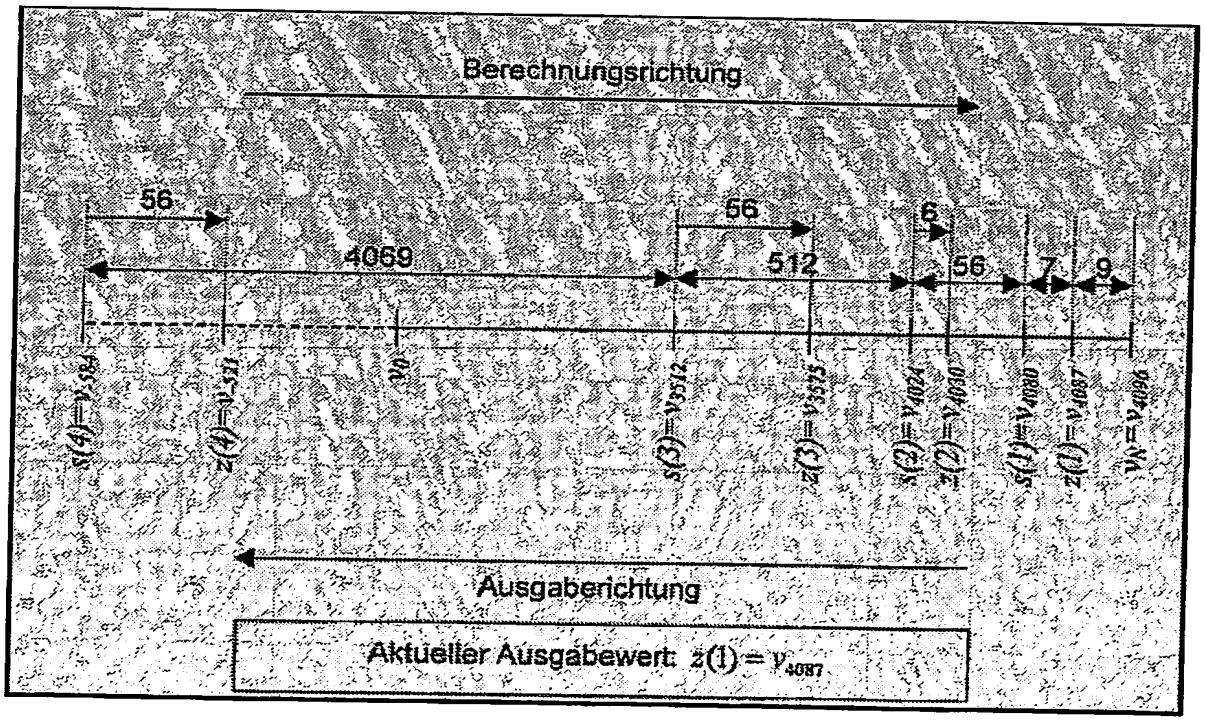
Berechnungsrichtung
56
4069
56
512
6
56
7
9
Ausgaberichtung
Aktueller Ausgabewert: $z(1) = v_{4087}$

Fig. 4

This Page is Inserted by IFW Indexing and Scanning Operations and is not part of the Official Record

BEST AVAILABLE IMAGES

Defective images within this document are accurate representations of the original documents submitted by the applicant.

Defects in the images include but are not limited to the items checked:

- ☐ BLACK BORDERS
- ☐ IMAGE CUT OFF AT TOP, BOTTOM OR SIDES
- ☐ FADED TEXT OR DRAWING
- ☐ BLURRED OR ILLEGIBLE TEXT OR DRAWING
- ☐ SKEWED/SLANTED IMAGES
- ☐ COLOR OR BLACK AND WHITE PHOTOGRAPHS
- ☐ GRAY SCALE DOCUMENTS
- ☒ LINES OR MARKS ON ORIGINAL DOCUMENT
- ☐ REFERENCE(S) OR EXHIBIT(S) SUBMITTED ARE POOR QUALITY
- ☐ OTHER: _____

IMAGES ARE BEST AVAILABLE COPY.
As rescanning these documents will not correct the image problems checked, please do not report these problems to the IFW Image Problem Mailbox.